

רשתות מחשבים

פרק 2- תכנות socket

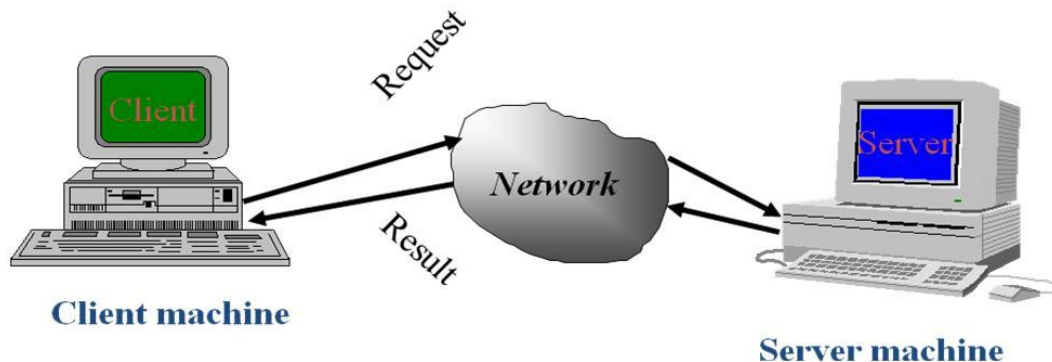
ברק גונן

מבוסס על ספר הלימוד "רשתות מחשבים" מאת

עומר רוזנבוים

מה נלמד בפרק זה?

- ▶ מהי תקשורת שרת - לקוח
- ▶ מהו socket
- ▶ נכתוב שרת ולקוח בשפת python, בעזרת socket
- ▶ השרת והלקוח יתקשרו ביניהם:
 - שליחת הודעות
 - העברת קבצים
 - תמונות
 - וכו'



תקשורת שרת-לקוח

- ▶ שרת - server, לקוח - client
- ▶ צורת התקשורת הנפוצה ביותר באינטרנט
- ▶ ה-server מספק שירות כלשהו
- ▶ ה-client פונה ל-server כדי להשתמש בשירות
- ▶ התקשורת בין השרת והלקוח מתבצעת על ידי socket



מהו socket?

- ▶ Socket הוא נקודת קצה של חיבור בין שני רכיבים
 - אם רוצים להעביר מידע בין מחשבים, צריך לקשר ביניהם
 - גם העברת מידע בין תוכנות שרצות על אותו מחשב מצריכה קישור
 - נקודות הכניסה והיציאה של המידע מכונות socket
- ▶ אפשר להמשיל socket לצינור:
 - ב-socket זורם מידע- זרם של בתים
 - זרימת המידע היא דו כיוונית
 - יש ל-socket נקודות התחלה וסיום



כתובות של socket

▶ כדי ש-socket יוכל לשמש להעברת מידע, צריך להגדיר את נקודות הקצה שלו

▶ הגדרת נקודת קצה מתבצעת ע"י שני מזהים:

- מזהה הרכיב- עם איזה מחשב מתקיימת התקשורת?
- מזהה התהליך- על המחשב שאיתו מתקיימת התקשורת רצות מספר תוכנות. עם איזו תוכנה מתקיימת התקשורת?
- Socket הוא צירוף של 2 נקודות קצה- שרת ולקוח

▶ מזהה הרכיב: כתובת IP

▶ מזהה התהליך: מספר פורט Port

- מספר בטווח 0-65535

▶ Socket מוגדר על ידי צירוף IP ו-Port



המחשה - IP ו-Port

- ▶ שרת שכתובת ה-IP שלו היא 1.2.3.4 מאחסן דפי אינטרנט שונים
 - השרת תומך בהעברת דפים רגילה HTTP או מאובטחת HTTPS
- ▶ שרת שכתובת ה-IP שלו היא 5.6.7.8 מספק מספר שירותים
 - השרת תומך בהעברת קבצים, יודע לקשר בין כתובות דומיין לכתובות IP ויודע לטפל באימיילים
- ▶ לאיזה צירוף של IP ו-Port צריך לפנות לקוח שרוצה לבצע תשאול DNS? גלישה מאובטחת לאתר אינטרנט?

תהליכים:

1. גלישה לדפי אינטרנט HTTP - פורט 80
2. גלישה לדפי אינטרנט מאובטחים HTTPS - פורט 443



שרת 1.2.3.4

תהליכים:

1. העברת קבצים FTP - פורט 20
2. מענה לבקשות DNS - פורט 53
3. שליחה וקבל אימיילים SMTP - פורט 25



שרת 5.6.7.8

שאלה למחשבה



- ▶ על השרת שכתובתו 1.2.3.4 הותקן דפדפן. תלמיד רוצה לגלוש מהשרת אל אתר אינטרנט, שמאוחסן על אותו השרת. לאיזה IP צריך לפנות?
- ▶ רמז: ה-IP אינו 1.2.3.4
- ▶ תשובה: 127.0.0.1. משמעות IP זה הינה "כתובת הבית", כלומר, התוכנה מולה נפתח ה-socket נמצאת באותו המחשב

- ▶ הלקוח מנסה להתחבר לשרת – Port, IP
- ▶ אם השרת מאזין בפורט הנ"ל, נוצר קשר חדש דו כיווני

AF_INET:
Internet
protocol (IP)

SOCK_STREAM:
Connection
type (TCP)

```
import socket

my_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
my_socket.connect(('127.0.0.1', 1729))

my_socket.send("Some Data To Send Here")
data = my_socket.recv(1024)

my_socket.close()
```


תרגיל כיתה – 2.2 לקוח לשרת הדים

▶ הורידו את השרת מהכתובת

http://cyber.org.il/networks/c02/echo_server_stream.pyc

▶ שימרו את הקובץ למיקום

C:\Cyber\echo_server_stream.pyc

▶ מה-command line הריצו את הפקודה:

```
python C:\Cyber\echo_server_stream.pyc
```

▶ כעת כיתבו לקוח ש:

- מתחבר אל השרת (פורט 1729)

- שולח הודעה אל השרת

- מקבל את תשובת השרת ומדפיס אותה על המסך

תרגיל כיתה – 2.3 כתיבת שרת

כיתבו שרת שמקבל את פניית הלקוח ועונה לו Hello ▶

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 1729))
server_socket.listen(1)

client_socket, address = server_socket.accept()

client_socket.send("Some Data To Send Here")
data = client_socket.recv(1024)

client_socket.close()
server_socket.close()
```

תרגיל כיתה – 2.3 כתיבת שרת

▶ כיתבו שרת שמקבל את פניית הלקוח ועונה לו Hello

0.0.0.0:
Listen to all
IP's on this
computer

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 1729))
server_socket.listen(1)

client_socket, address = server_socket.accept()

client_socket.send("Some Data To Send Here")
data = client_socket.recv(1024)

client_socket.close()
server_socket.close()
```

תרגיל כיתה - 2.3 כתיבת שרת

▶ כיתבו שרת שמקבל את פניית הלקוח ועונה לו Hello

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 17)
server_socket.listen(1)

client_socket, address = server_socket.accept()

client_socket.send("Some Data To Send Here")
data = client_socket.recv(1024)

client_socket.close()
server_socket.close()
```

Define how many clients can wait for connection

תרגיל כיתה – 2.3 כתיבת שרת

▶ כיתבו שרת שמקבל את פניית הלקוח ועונה לו Hello

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 1729))
server_socket.listen(1)

client_socket, address = server_socket.accept()

client_socket.send("Some Data To Send Here")
data = client_socket.recv(1024)

client_socket.close()
server_socket.close()
```

Wait for client
connection

תרגיל כיתה - 2.3 כתיבת שרת

▶ כיתבו שרת שמקבל את פניית הלקוח ועונה לו Hello

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 1729))
server_socket.listen(1)

client_socket, address = server_socket.accept()

client_socket.send("Some Data To Send Here")
data = client_socket.recv(1024)

client_socket.close()
server_socket.close()
```

client_socket:
Communication
with a specific
client

תרגיל כיתה - 2.3 כתיבת שרת

כיתבו שרת שמקבל את פניית הלקוח ועונה לו Hello ▶

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 1729))
server_socket.listen(1)

client_socket, address = server_socket.accept()

client_socket.send("Some Data To Send Here")
data = client_socket.recv(1024)

client_socket.close()
server_socket.close()
```

Note: this is a
tuple

תרגיל כיתה- 2.4 הרצת שרת ולקוח

פיתחו שני חלונות command line ▶

◦ בראשון הריצו את server.py

◦ בשני הריצו את client.py

▶ אפשר להשתמש ב-Pycharm, מומלץ כשרוצים לדבג

```
C:\Windows\system32\cmd.exe - server.py
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\USER>cd \Cyber
C:\Cyber>server.py
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\USER>cd \Cyber
C:\Cyber>client.py
The server sent: Hello Omer
C:\Cyber>
```

תרגיל 2.5- שרת הדים

- ▶ בתרגיל 2.2 כתבתם לקוח שהתחבר לשרת קיים
- ▶ כעת כיתבו את השרת
 - משכפל כל מידע שנשלח אליו ושולח חזרה
- ▶ טיפ: לבצע `close` בסוף התוכנית
 - משחרר את משאבי המערכת ובעיקר את הפורט
- ▶ טיפ: מומלץ לעבוד עם `pycharm` במצב דיבוג, כך שאם התוכנית קורסת לפני הגעה ל-`close`, ניתן לשחרר את הפורט (עצירת הדיבוג)

פרוטוקול תקשורת - תזכורת

- ▶ פרוטוקול Protocol הוא אוסף של חוקים שמאפשר לשתף ישויות או יותר להעביר ביניהן מידע
- ▶ דוגמה:



תרגיל 2.6 – שרת פקודות בסיסי

▶ השרת מבצע פקודות שהלקוח שולח ומחזיר תשובה

▶ רשימת הפקודות:

◦ Time, Name, Rand, Exit (פירוט בספר)

▶ טיפים לביצוע התרגיל:

◦ קלט משתמש- פקודת raw_input

◦ הלקוח שולח כמות בתים קבועה- socket.send(4)

◦ השרת מחזיר כמות בתים משתנה

• ניתן לרפד את תשובת השרת באפסים וכך להגיע לאורך קבוע

• ניתן לכתוב בתחילת תשובת השרת את כמות הבתים בהמשך

◦ תכננו מראש את פרוטוקול התקשורת בין השרת ללקוח

תרגיל 2.7- שרת פקודות מתקדם

- ▶ בתרגיל זה תכתבו שרת-לקוח של תוכנה שמאפשרת לטכנאי לבצע פעולות שונות על מחשב מרוחק:
 - קבלת צילום מסך
 - הפעלת תוכנות שונות
 - העתקה של קובץ מהשרת
 - הצגת תוכן תיקיה, מחיקת קבצים
- ▶ הדרכה נמצאת בספר הלימוד ובסרטונים הבאים:
 - [הנחיות לתרגיל 2.7](#)
 - [תרגיל 2.7 דוגמת הרצה](#)
- ▶ טיפים לעבודה- בשקף הבא

תרגיל 2.7- הדרכה וטיפים

- ▶ תכננו מראש את פרוטוקול התקשורת בין השרת והלקוח
- ▶ תכננו איך הלקוח יודע שהעברת הקובץ הסתיימה?
 - `socket.send(message)` לא שולחת כלום אם `message` ריק
- ▶ בעבודה עם קבצים, יש סיכון שהלקוח יבקש לבצע פעולה על קובץ לא קיים
 - העזרו בתנאים `try, except` כדי למנוע קריסה
- ▶ כדי לעבוד עם `subprocess.call`, יש צורך להעביר כפרמטר את המיקום המלא של התוכנה שתופעל

תרגיל 2.7- הדרכה לעבודה עם pil

▶ שלב א'- היכנסו לקישור

<http://www.pythonware.com/products/pil/>

▶ שלב ב'- הורידו את pil לגרסת פייתון 2.7 לחלונות
והתקינו

▶ במידה וההתקנה נכשלת, פיתחו cmd והקלידו:

```
pip install PIL
```

▶ אם זה נכשל, היכנסו לספרייה הבאה ונסו שוב להקליק
את `pip install PIL`:

```
cd c:\Heights\PortableApps\PortablePython2*\App\Scripts
```

סיכום- מה למדנו?

- ▶ מודל עבודה שרת - לקוח
- ▶ כתיבת socket
- ▶ פיתוח שרת ולקוח ב-python
- ▶ תכנון פרוטוקול תקשורת

